Zheen H. Suseyi

02/25/2024

CS 470

SNHU

Project Two Conference Presentation: Cloud Development

https://www.youtube.com/watch?v=CM7b0w6iHxo

Hello my name is Zheen Suseyi and I am a 4th year Computer Science Student at Southern New Hampshire University. The purpose of this presentation is to articulate the intricacies of cloud development to both technical and nontechnical audiences.

In this project, I turned a full stack locally hosted application to a full stack cloud hosted application on AWS. The project started as a docker container hosted locally on localhost3000 and localhost4200. Eventually, a static web application was hosted on S3 and from there every component of AWS was utilized in order to host the full stack application on the cloud.

The tools that are necessary for containerization would be docker, docker compose and orchestration, and creating docker images.

Using Docker Compose allows for the ability to merge multiple docker containers at once and to make them work in unison. During the project, Docker Compose and Orchestration was utilized in order to add additional functionality to the locally hosted application.

This project utilized AWS infrastructure in order to host the application on the cloud. This is also known as serverless because the application is no longer locally hosted and instead, hosted by AWS. The two major advantages of serverless include much greater scalability and reduced cost for larger applications.

S3 storage is a service provided by AWS that allows for creating buckets that can store objects and applications. Compared to local storage, S3 definitely has some advantages because it utilizes AWS infrastructure which allows for rapid scalability, security, and cost effectiveness.

The advantages of using a serverless API include rapid scalability, cost effectiveness, security, and the ability to utilize the AWS ecosystem. AWS Lambda API is a serverless function that allows a user to run code without provisioning or managing servers (1). The scripts produced in order to make this happen was firstly creating a Lambda function, editing index.js, creating test cases, then finally testing and deploying the code.

From Modules 3-6, the focus of the project was to deploy the frontend and backend using AWS services. In Module 7, both the frontend and backend were combined in order to host the full application on AWS, the steps can be summarized here:

1. Create The Questions API

2. Create The Answers API

3. Adding Cross-Origin Resource Sharing to both the Questions and Answers API

4. Modifying and deploying the Angular Application

The AWS database is called DynamoDB and this is different from the commonly known MongoDB. The data-model differences between the two can be boiled down to MongoDB storing its data using a JSON based document store while DynamoDB has a Limited key-value store with JSON support (2). The queries that were performed during the project were 5 Lambdas. TableScan, GetSingleRecord, UpsertQuestion, UpsertAnswer, and DeleteRecord. These are the five Lambdas that were created and each of them had test cases in order to make sure they function.

Because the project is hosted using AWS infrastructure, elasticity isn't an issue because of the scalability that AWS offers. A feature of AWS includes a Pay-for-use model where you only pay for what your application actually utilizes. This chart demonstrates how a Traditional Data Center can fail a team and lead to Customer Dissatisfaction due to too much usage or vice versa.

Preventing unauthorized access was done through the roles given by SNHU. It was impossible to create a Lambda function unless a role called LabRole was used which had the appropriate policies attached to it. This chart shows how IAM can be assigned to certain groups of users to give them the necessary permissions.

The relationship between roles and policies during this project was that certain functionalities like creating a Lambda was not possible unless the user was assigned the "LabRole". The reason why users who were assigned LabRole could create Lambas was because there were policies attached to LabRole that allowed for that. The custom policies that were created were already assigned to LabRole and LabRole didn't allow for any policy creation as shown in the image.

Assigning the correct IAM roles to the correct users is key when securing the connection between Lambda and Gateway, Lambda and the database, and S3 Bucket. For S3 Bucket in particular, it is important to set up Bucket Policies and Permissions as well like in the image shown.

This project involved moving a full-stack application from my local system to the AWS cloud. There are many benefits to cloud development but if I were to focus on three main points they would be the following.

1. AWS cloud offers reliable scalability that only makes the user pay for what they use.

2. Cloud Development can be made extremely secure using IAM roles and policies.

3. A static website on the cloud can easily be turned into a full-stack application that is accessible to anyone.

Thank you for your time!